Unit Preference for Ordered Resolution and for Connection Graph Resolution

Reiner Hähnle[1], Neil V. Murray[2], Erik Rosenthal[3]

[1] Chalmers University of Technology, Department of Computing Science, 41296 Gothenburg, Sweden, reiner@cs.chalmers.se
[2] Inst. for Informatics, Logics, & Security Studies, Dept. of Comp. Sci., University at Albany – SUNY, Albany, NY 12222, USA, nvm@cs.albany.edu
[3] Department of Mathematics, University of New Haven, West Haven, CT 06516, USA, erosenthal@newhaven.edu

Abstract. Strong completeness, which is almost automatic at the ground level for the tableau method and for path dissolution, is a question for many path-based calculi, most notably for *cg*-resolution. Because of the destructive nature of *cg*-resolution, many results, including completeness, require non-trivial proofs; others hold only if the calculus is restricted. Norbert Eisinger's famous example showed that unrestricted cg-resolution is not strongly complete. One restricted form of *cg*-resolution — ordered resolution — is.

In this paper, a relatively simple proof of the ground confluence of unrestricted *cg*-resolution is presented and strengthened to exclude tautology creation. In addition, ordered *cg*-resolution (and ordered set resolution) are shown to be compatible with a unit preference strategy, while preserving strong completeness.

Keywords: *Completeness, strong completeness, connection-graph resolution, ordered resolution, unit preference, subsumption, link deletion.*

1 Introduction

Strong completeness — carefully defined in Section 2 but, roughly speaking, an inference system is strongly complete if any sequence of steps eventually terminates — is almost automatic at the ground level for the tableau method and for path dissolution, but it is a question for many path-based calculi, most notably for *cg*-resolution. Because of the destructive nature of *cg*-resolution, many results, including completeness, require non-trivial proofs; others hold only if the calculus is restricted. Norbert Eisinger's [9] famous example showed that unrestricted cg-resolution is not strongly complete. One restricted form of *cg*-resolution — ordered resolution — is (see, for example, [11]).

Refutation confluent, non-destructive, complete calculi combined with appropriate fairness strategies are always strongly complete. This is also true for a complete destructive calculus, but the appropriate fairness strategy may entail substantial backtracking. When inference steps are complex as in cg-resolution

(as opposed to Prolog), backtracking is likely to be expensive and thus eliminate the search space advantages of a destructive calculus.

Connection graph resolution (*cg*-resolution) was introduced by Kowalski [15] in 1975. It is a modification of resolution in which links are deleted upon activation, thus restricting the search space. The possiblity of a substantial reduction in the size of the search space led to extensive research into its properties. Some of the early work included Brown [7], who investigated a variant of the system based on semantic trees, and Siekmann and Stephan [20], who studied a related idea using a minimal unsatisfiable kernel. But investigators quickly discovered that the behavior of *cg*-resolution is rather elusive. Determining and verifying its properties has been fraught with difficulties, some of which will be explored in this paper.

Most researchers assumed *cg*-resolution to be complete and hoped it was strongly complete, but it was six years before completeness was actually proved. Both the semantic tree and excess literal techniques seemed to be insufficient, and the first proof did not appear until Bibel's seminal paper [4] in 1981. The crucial advance he made was introducing the notion of spanning and observing that *cg*-resolution steps preserve spanning (see Section 3). His proof requires that, throughout a refutation, the connection graph is "almost" fully connected, primarily to ensure that tautologies may always be deleted. But he did not prove *confluence*; i.e. his proof admits the possibility that a sequence of link activations will yield a connection graph for which a proof is no longer available. Bibel [5] established completeness for the first-order case, but only if the "copy-rule" is available. The graphs are again almost fully connected, and derivations are constructed so that tautologies are removable.

Several of Siekmann's students attacked the problem. Smolka [24] established first-order confluence for the unit-refutable class with tautology deletion.[1] Eisinger [10] provided an intricate collection of results for the first-order case that imply ground confluence (without mention of tautology avoidance).

In Section 3.2, a variation of the Anderson-Bledsoe [1] excess literal technique is employed to prove that *cg*-resolution is complete. It is simpler than the proof in Bibel's 1981 paper, much simpler than Eisinger's construction, and requires only a spanning set of links. Confluence is an immediate consequence. This result is generalized in Section 3.3 to show that tautologies need never be created (but not that they can always be deleted if they are already present!).

More than one author[2] has tried to prove strong completeness of *cg*-resolution. There were several failed attempts at proving strong completeness for unrestricted *cg*-resolution before Eisinger's famous counterexample [9, 10] dispelled

[1] Very recent discussions with Siekmann and with Eisinger, for which the authors are grateful, have revealed that in [25], Smolka developed ground confluence for the non unit-refutable case. The results are unpublished, appearing only in German in his diploma thesis. We believe the development here to be useful, not only to give these insights wider accessability, but also because they are carefully crafted to support the main results on unit preference.

[2] The intersection of that club with the authors of this paper is non-empty.

that notion. He showed that even a quite restrictive concept of fairness cannot prevent cyclic derivations. Investigations with other restrictions that might admit strong completeness have generally met with failure. This is in contrast to some calculi — the tableau method and path dissolution, for example — that also delete links (either explicitly or implicitly) but are easily seen to be strongly complete.

Eisinger's example is a bit complicated: It cycles after sixty-six steps. In 1987, Bibel [5] found a simpler example. He noted that a fairness condition would eliminate his example but not Eisinger's. Resolution with free selection function is not complete, and any example that demonstrates its incompleteness is easily seen also to be a counterexample to strong completeness of unrestricted *cg*-resolution.

Ordered resolution predates *cg*-resolution and is attributed to Reynolds [18]. Variations have been proven complete by Kowalski and Hayes [13], by Joyner [12] and by Bachmair and Ganzinger [2]. Bachmair and Ganzinger also proved strong completeness. Ordered resolution is the basis of modern saturation-based calculi that are used in many successful automated deduction systems. Ordered resolution is related to *cg*-resolution insofar as it was shown to be a special case of *cg*-resolution in [11]. This proves strong completeness for ordered *cg*-resolution. In Section 3.4, that result is generalized to admit activation of links to unit clauses *even when unordered.* Noteworthy is the inclusion of *cg*-subsumption by units and by unit resolvents (but not in general). This result in turn implies that ordered resolution with unordered unit link activations is strongly complete, providing, at least as far as the authors are aware, the first combination of unit preference with an ordering restriction.

Attention is restricted to the ground case in this paper; some remarks about the first-order level are made at the end.

2 Strong Completeness

To properly define strong completeness, it is important to be clear on the distinction between *inference system* or *calculus*, and *proof procedure.* Kowalski [14] was perhaps the first to draw this distinction carefully in the deduction literature. The former is a formal language along with rules of inference that define admissible operations on formulas of that language. Thus, given any set of formulas and an inference system, there is a (usually finite) set of admissible inference steps that can be applied to produce a new set of formulas.

A proof procedure, on the other hand, is an inference system combined with a search strategy. A search strategy is a deterministic process that, when presented with the set of admissible steps for a given formula set, chooses the next step to be performed. If a (refutation based) proof procedure is complete, then it terminates after a finite number of steps when applied to an unsatisfiable set of formulas; i.e., a complete proof procedure is automatically "strongly complete."

An inference system, or calculus, is *strongly complete* if every proof procedure consisting of that calculus and an arbitrary search strategy is complete. In other

words, termination is guaranteed even if, at each step, the choice from among admissible steps is arbitrary.[3]

The confusion that often arises with the notion of strong completeness is usually due to ambiguities regarding "restrictions" or "refinements" of various inference rules. One must first decide whether the calculus includes such restrictions, or if instead they are part of the search strategy.

One can always start with a complete proof procedure and embed the search strategy into the formal definition of the original calculus to produce a strongly complete calculus. The question of whether strong completeness is interesting for the resulting calculus is subjective. On the other hand, establishing strong completeness for a destructive calculus is typically desirable but often difficult.

In [10], Eisinger characterizes *cg*-resolution with respect to several properties including *refutation soundness*, *refutation completeness*, and *refutation confluence*. Informally, refutation soundness means that the empty clause can be derived only from initial graphs that are unsatisfiable. Refutation completeness means that for every unsatisfiable initial connection graph there exists a refutation. Refutation confluence means that if G is an unsatisfiable initial graph and both G_1 and G_2 can be derived from G, then there is a graph G' such that G' is derivable from both G_1 and G_2. Non-destructive calculi are automatically confluent, and thus refutation completeness is the issue of primary interest.

For *cg*-resolution, refutation soundness is inherited from ordinary resolution and is immediate.[4] But because *cg*-resolution operates destructively and only on links that are present, unsatisfiability alone is insufficient for discussion of properties such as confluence. The sentence $\{\{p\}, \{\overline{p}\}\}$ is unsatisfiable but not refutable if the associated link set is empty. The crucial point is that the existence or inevitability of refutations from a given state must be analyzed with respect to properties of the link set as well as with respect to unsatisfiability.

Since *cg*-resolution is complete, it may be combined with backtracking to form a strongly complete proof procedure. But the base calculus is destructive, and a single backtracking step would require "unresolving" — i.e., restoration of a deleted link, the deletion of the resolvent and its links, and the restoration of any clauses that were removed by purity. To say the least, this would be esthetically distasteful, not to mention enormously inefficient. (Of course, backtracking in Prolog is efficient because it amounts to retreating from one simple step in a linear proof.)

Another example of a strongly complete strategy is Bibel's original proof of the completeness of *cg*-resolution. It not only provided the first completeness proof for the basic calculus but also gave a terminating search strategy. Predicates are ordered, and only links of maximal order are selected by the search strategy. (The set of maximal links is different from the set of ordered links — see Section 3.4.) Declaring activations of maximal links to be the only admissible

[3] Repeated identical steps are implicitly disallowed, although this is typically not possible with a destructive calculus.

[4] This is in general not the case for the *cg*-purity rule, because a clause that is pure in the *cg* sense is not necessarily pure in the ordinary sense.

steps defines a calculus, and termination is assured in the absence of a search strategy. However, this calculus forces a very large space of partial proofs, many of which are eventually one step away from a refutation long before that one step is carried out.

3 Ordered and Connection Graph Resolution

A *multiset* over a set $\mathcal{L}$ is a mapping M from $\mathcal{L}$ to the non-negative integers; M is finite if $M(l) = 0$ for all but finitely many $l \in \mathcal{L}$. Conceptually, a multiset is a collection of any number (including zero) of occurrences of the elements of $\mathcal{L}$. The set $\mathcal{L}$ may be treated as a multiset by letting $M(l) = 1$ for all $l \in \mathcal{L}$. If $\prec$ is a *strict partial order* (i.e., an irreflexive, transitive relation) on $\mathcal{L}$, then $\prec$ can be extended to multisets. If M and M' are distinct multisets over $\mathcal{L}$, then $M' \prec M$ if whenever there is an l in $\mathcal{L}$ with $M(l) < M'(l)$, there is an l' in $\mathcal{L}$ with $l \prec l'$ and $M'(l') < M(l')$. In other words, to diminish a multiset, remove one occurrence of an element and replace it with any finite number of occurrences of smaller elements. Well-foundedness and totality of $\prec$ are inherited by a (finite) multiset extension; a good source for this material is [8].

Assume a countable propositional *signature* (i.e., atom set). Since a clause may be treated as a multiset of literals, any total order on literals defines an order on clauses. The largest literal in each clause is called the *head* of the clause. If each atom immediately precedes its negation in the literal ordering, then the ordering is said to be an *atom ordering.*

To define ordered and connection graph resolution, it is convenient to begin with a definition of ordinary resolution. A *link* in a set of clauses $\mathcal{S}$ is a set $\{l, \overline{l}\}$, where the complementary literals l and $\overline{l}$ occur in distinct clauses of $\mathcal{S}$. An occurrence of the literal l in the clause C may be written l_C, and $\{l_C, \overline{l}_D\}$ denotes the link $\{l, \overline{l}\}$ for which $l \in C, \overline{l} \in D$. Then the clause $E = (C - \{l_C\}) \cup (D - \{\overline{l}_D\})$ is the *resolvent* of $\mathcal{S}$ with respect to the link $\{l_C, \overline{l}_D\}$; the link is said to be *activated.* A link between heads of clauses is called an *ordered link.* If an ordered link is activated, its resolvent is said to be *ordered*, and the resolution procedure that requires every resolvent to be ordered is called *ordered resolution.*

Links are sets of ordered literals and hence are ordered as well. This order can in turn be extended to multisets of links. All orderings are well-founded because the number of distinct clauses and links is finite.

A *resolution derivation* of a clause E from a clause set $\mathcal{S}$ is a finite sequence of clauses $D_1, D_2, \ldots, D_n = E$ such that D_i is a resolvent of $\mathcal{S} \cup \{D_1, \ldots, D_{i-1}\}$ for all $1 \leq i \leq n$. Such a derivation may also be viewed as the sequence of clause sets $\mathcal{S}, S \cup \{D_1\}, S \cup \{D_1, D_2\}, \ldots$. An *ordered resolution derivation* is one in which each resolution step is ordered.

Connection graph resolution is essentially ordinary resolution with link deletion. As a result, it is necessary to keep track of which links are present. Thus, a *connection graph* (*c-graph*) is defined to be a pair $(\mathcal{S}, \mathcal{L})$, where $\mathcal{S}$ is a finite multiset of clauses, and $\mathcal{L}$ is a set of links in $\mathcal{S}$. The clauses of a c-graph are technically a multiset because identical copies of clauses may have different associated links. As usual, $\Box$ denotes any c-graph whose clause set contains the

empty clause.[5] The connection graph containing all possible links is called the *full graph* of the clause set $\mathcal{S}$.

The notions of path and spanning are key to *cg*-resolution. A *conjunctive path* or *c-path* through a clause set $\mathcal{S}$ is a set of literal occurrences containing exactly one literal from each clause in $\mathcal{S}$. A *c*-graph $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ is said to be *spanned* by $\mathcal{L}$ if $\mathcal{S} \neq$ and each c-path through $\mathcal{S}$ contains a link from $\mathcal{L}$; it is *minimally spanned* if removal of any clause and its associated links produces a graph that is not spanned. Observe that a spanned formula is unsatisfiable, and a formula is unsatisfiable if and only if it is spanned by the full set of links. Note, however, that a link set $\mathcal{L}$ may span $\mathcal{S}$ and yet not be full. Intuitively, if $\mathcal{L}$ spans $\mathcal{S}$, then $\mathcal{L}$ contains enough information to demonstrate the unsatisfiability of $\mathcal{S}$.

Example. Consider the *c*-graph containing the clauses $A = \{p, q\}$, $B = \{\overline{p}, \overline{q}\}$, $C = \{\overline{p}, q\}$, $D = \{p, \overline{q}\}$ and containing all links except the two between A and B. Observe first that the clause set is minimally unsatisfiable; that is, any three of the clauses is satisfiable. Yet the *c*-graph is spanned but not full.

To define connection graph resolution, let $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ be a connection graph, let $L = \{l_C, \overline{l}_D\}$ be a link in $\mathcal{L}$, and let E be the clause obtained by resolving on L. As indicated by our notation, we denote by l'_E a literal l' in the resolvent E that was present in C (l'_C) or in D (l'_D) or in both. Then *connection graph resolution* on the link L produces the connection graph $\mathcal{G}' = (\mathcal{S}', \mathcal{L}')$, where $\mathcal{S}' = \mathcal{S} \cup \{E\}$, and

$$\mathcal{L}' = (\mathcal{L} - \{L\}) \cup \{(l'_E, \overline{l'}_F) \mid (l'_C, \overline{l'}_F) \in \mathcal{L} \text{ or } (l'_D, \overline{l'}_F) \in \mathcal{L})\} \ .$$

The clause E is called a *cg-resolvent*. The links added to $\mathcal{L}$ to obtain $\mathcal{L}'$ are called *inherited links*. The definition above can be referred to as *full inheritance* — all links to parent literals are inherited. Spanning can be preserved under more restrictive inheritance rules, but for the remainder of this paper full inheritance will be assumed.

If $\mathcal{G} = \mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_n = \mathcal{H}$ is a sequence of *c*-graphs such that each $\mathcal{G}_i$ is produced from $\mathcal{G}_{i-1}$ by *cg*-resolution, then $\mathcal{H}$ is said to be obtained from $\mathcal{G}$ by a *cg-resolution derivation.*

Let us emphasize that what distinguishes *cg*-resolution from ordinary resolution is, given a *c*-graph $\mathcal{G} = (\mathcal{S}, \mathcal{L})$, only the links in $\mathcal{L}$ may be *cg*-resolved upon. A key property of *cg*-resolution is that each step preserves spanning.

Lemma 1 (*Spanning Lemma).* If a link in a spanned connection graph is *cg*-resolved upon, then the resulting connection graph is spanned. □

A literal occurrence l in a *c*-graph $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ is called *pure* in $\mathcal{G}$ if l is unlinked; i.e., if no link in $\mathcal{L}$ contains the occurrence l. The clause C is *pure* in $\mathcal{G}$ if it contains a pure literal. A clause that contains complementary literals is called a *tautology.* Pure clauses and tautologies are of interest because of the *Pure Rule* and the *TAUT Lemma*, which delete such clauses under some circumstances. The next lemma (and Lemma 1) are from Bibel's 1981 paper [4].

[5] In the presence of a subsumption rule, this graph can be assumed to be $(\{\Box\}, \emptyset)$.

Lemma 2 (*TAUT Lemma*). If $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ is a spanned connection graph, if $C \in S$ is a tautology containing the literal occurrences l_C and $\overline{l}_C$, and if C together with the associated links are deleted from $\mathcal{G}$, then the resulting connection graph is spanned provided that the following condition is met:

$$\{\{l_D, \overline{l}_E\} \mid \{\{l_D, \overline{l}_C\}, \{l_C, \overline{l}_E\}\} \subseteq \mathcal{L}\} \subseteq \mathcal{L} \, .$$

□

3.1 The Pure Rule and Subsumption

The next lemma is the Pure Rule for connection graphs [4]; its proof of spanning preservation is essentially identical to a proof of unsatisfiability preservation for the Pure Rule of ordinary resolution and is straightforward.

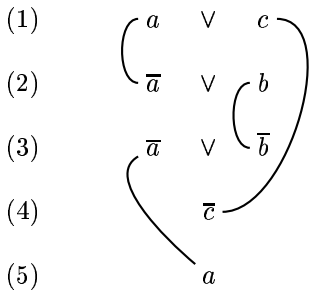
Lemma 3 (Pure Rule). Let $(\mathcal{S}, \mathcal{L})$ be spanned and contain the clause $C = A \cup \{l_C\}$, where l_C is pure. Then $(\mathcal{S}', \mathcal{L}')$ is spanned, where $\mathcal{S}' = \mathcal{S} - \{C\}$ and $\mathcal{L}'$ is the result of removing from $\mathcal{L}$ all links that meet clause C. □

In the next lemma, and in subsequent developments, objects are removed from the clauses of a connection graph. In such situations, it is implicitly assumed that links associated with the removed objects are also removed. The lemma is from [11].

Lemma 4. Let $(\mathcal{S}, \mathcal{L})$ with $C \in \mathcal{S}$ be a minimally spanned c-graph, and suppose $C = \{p\} \cup \{q_1, \ldots, q_n\}$, where $n \geq 0$. Obtain $\mathcal{G}' = (\mathcal{S}', \mathcal{L}')$ from $\mathcal{G}$ by deleting p from C. Then $\mathcal{G}'$ is spanned, and every minimally spanned subset of $\mathcal{S}'$ contains $C' = \{q_1, \ldots, q_n\}$ but contains no clauses that contain an unlinked $\overline{p}$. More generally, p may be removed from all clauses in $\mathcal{S}$ to produce a spanned c-graph with one fewer atom. □

The classical notion of subsumption carries over to the connection graph setting in a spanning-preserving manner [4]. However, it must be modified because link sets may not be full. The modification is straightforward. Suppose that the connection graph $(\mathcal{S}, \mathcal{L})$ contains clauses C and D, and that $C \subseteq D$; i.e., C "classically" subsumes D. For each literal l_C in C, let Lit_{l_C} denote the set of literal occurrences linked to l_C in $\mathcal{L}$. Suppose further that for every literal l_C, $Lit_{l_C} \supseteq Lit_{l_D}$. Then the clause C is said to *cg-subsume* clause D. The requirements on the links are necessary to ensure that spanning is preserved when a subsumed clause is removed.

Consider the following example:



Clause (5) does not cg-subsume (1). Indeed, if (5) is removed, subsequent purity deletions lead to the (satisfiable) graph in which the clause set is empty.

Lemma 5 (Cg-subsumption). Suppose that $(\mathcal{S}, \mathcal{L})$ is spanned and contains clauses C and D, and suppose that C *cg*-subsumes D. Then $(\mathcal{S}', \mathcal{L}')$ is spanned, where $\mathcal{S}' = \mathcal{S} - \{D\}$, and $\mathcal{L}'$ is the result of removing from $\mathcal{L}$ all links that meet clause D. □

3.2 Completeness

Bibel's 1981 paper introduced spanning and proved completeness; it was an important step forward in understanding connection graph resolution. The new proof presented here fills in some non-trivial gaps for the ground case that have persisted and that are outlined in [9, 10]. Although ground confluence can be deduced from [24, 10], the new proof is direct and is easily strengthened to prohibit tautology creation.

The induction in Theorem 1 below requires Lemma 6 (a proof of which can be found in [11]). The reason is that proof steps known to exist for a clause set $\mathcal{S}'$ must be applied to a larger set $\mathcal{S}$. Any link used in refuting $\mathcal{S}'$ must therefore be present when needed in $\mathcal{S}$. This is initially the case but could potentially fail if the Pure Rule initiated a "cascade" of purity deletions that remove an essential link from $\mathcal{S}$ before the link is activated. Such a cascade cannot occur in the ground case, because clauses become pure only from links deleted after activation. Intuitively, for each link removed from a non-parent clause because a parent becomes pure, there is an inherited link to the resolvent.

Lemma 6. A ground clause may become pure only from a *cg*-resolution step activating the sole link to one of its literals. □

Theorem 1. Connection-graph resolution is refutation complete for spanned ground connection graphs. In particular, since *cg*-resolution steps preserve spanning, *cg*-resolution is confluent.

Proof. Let $(\mathcal{S}, \mathcal{L})$ be a connection graph spanned by $\mathcal{L}$; we must show that there is a refutation of $\mathcal{S}$ using *cg*-resolution. We proceed by induction on the number n of distinct atoms in $\mathcal{S}$. If there are none, then $\mathcal{S}$ contains the empty clause, and we are done. Otherwise, assume that all spanned connection graphs with at most n atoms can be refuted with *cg*-resolution, and suppose that $\mathcal{S}$ has $n + 1$.

If $\mathcal{S}$ is not minimally spanned, then restrict attention to a minimally spanned subset. Let p be any atom in $\mathcal{S}$; we begin by deriving the unit clause $\{p\}$ from $\mathcal{S}$. If $\mathcal{S}$ contains $\{p\}$, we have it; otherwise, by Lemma 4, we can remove all occurrences of p in $\mathcal{S}$, producing the spanned connection graph $(\mathcal{S}', \mathcal{L}')$. By the induction hypothesis, there is a refutation of $\mathcal{S}'$. By Lemma 6, this refutation can be applied[6] to $\mathcal{S}$ and produces either the empty clause, in which case the proof is complete, or the unit clause $\{p\}$, as promised.

[6] Saying that the refutation of $\mathcal{S}'$ is "being applied" to the clause set $\mathcal{S}$ means that each resolution step involving a clause $C' \in \mathcal{S}'$ is applied to the clause $C \in \mathcal{S}$ that produced C'. This is always possible, because $C' \subseteq C$.

The unit $\{p\}$ cannot be pure because no occurrence of p in $\mathcal{S}$ was pure, and no resolution step involved any clause from $\mathcal{S}$ that contained $\overline{p}$. In particular, there are no new clauses containing $\overline{p}$. Let $C = \{\overline{p}\} \cup C'$ be a clause that is linked to p. The *cg*-resolvent of $\{p\}$ and C is C', which *cg*-subsumes C, since every link to C' is inherited directly from a link to C. By activating all links to $\{p\}$, a minimally spanned subset of the resulting connection graph is a spanned clause set in which the number of occurrences of $\overline{p}$ has been reduced (but not necessarily to zero, because there might be occurrences of p other than in the unit clauses $\{p\}$).

This process can be iterated. Each iteration produces a unit $\{p\}$ (or the empty clause). Links to the unit $\{p\}$ are activated, reducing the number of occurrences of $\overline{p}$, while preserving spanning. Since the number of occurrences of $\overline{p}$ is reduced with each iteration, this process terminates. Eventually a spanned graph with no occurrences of $\overline{p}$ is produced. By the Pure Rule, a minimally spanned subset contains no occurrences of p. Since this minimally spanned subset has at most n distinct atoms, by the induction hypothesis, there is a refutation with *cg*-resolution. □

Note that the notion of *cg*-subsumption is employed in the proof above, but not as part of the inference machinery. The unit *cg*-resolutions between the derived unit $\{p\}$ and clauses containing $\overline{p}$ produce *cg*-subsuming clauses. The existence of minimally spanned clause sets that do not contain the parent clauses with $\overline{p}$ is the key point. No claim was made that the subsumed clauses were in fact deleted.

It is not at all clear that Theorem 1 can be strengthened to include *cg*-subsumption. Lemma 6, which guarantees the absence of cascading purity, is crucial for the application to $\mathcal{S}$ of the refutation of $\mathcal{S}'$. But *cg*-subsumption, unlike *cg*-resolution, only deletes links; no new links that can prevent a cascade effect are inherited once a subsumed clause is deleted. Suppose for example that, when applying the refutation of $\mathcal{S}'$ to $\mathcal{S}$, a clause is produced that *cg*-subsumes a clause $R = \{\overline{p}\} \cup E$. In $\mathcal{S}'$, R was never resolved upon. However, if it were deleted in $\mathcal{S}$ by *cg*-subsumption, it is possible that a clause in $\mathcal{S}$ containing p and used in the proof could become pure.

3.3 Tautologies

Tautologies provide another example of the subtleties of the *cg*-resolution calculus. Ideally, they would simply be removed, which in turn would imply they need never be created. However, the TAUT Lemma may not apply to a spanned graph that is not fully linked, in which case tautologies cannot be removed if spanning is to be maintained. However, Theorem 2 below, which holds for any spanned graph, does guarantee that the creation of new tautologies can be avoided, even if existing ones might be required for a proof.

There are two ways to create a tautology. To understand how, consider the following example. If we resolve $\{A, B, \overline{D}\}$ with tautology $\{\overline{E}, E, D\}$ we produce a new tautology $\{A, B, \overline{E}, E\}$ *by copy*. If instead we resolve $\{A, B, \overline{D}\}$ with $\{\overline{A}, D, E\}$ we produce a new tautology $\{B, \overline{D}, D, E\}$ *by link*. Tautologies that are

already present may not always be avoided. As a result, it is not terribly interesting to avoid tautology creation by copy. Tautology creation by link, however, is avoidable.

Suppose we add to the hypotheses of Theorem 1 the assumption that links whose activations would produce tautologies by link are never selected. It is easy to adjust the proof of that theorem to accommodate the additional hypothesis. Obviously the base case is unaffected, and the induction hypothesis produces a refutation of $\mathcal{S}'$ by *cg*-resolution in which no tautology is created by link. Clauses in the refutation of $\mathcal{S}'$ do not contain $\overline{p}$. Since adding p back into any such clause cannot create a tautology, the derivation of the unit $\{p\}$ is tautology free. Generating the *cg*-resolvents of the unit $\{p\}$ with clauses containing $\overline{p}$ also will not produce a tautology by link. Therefore, as in the proof of Theorem 1, a spanned graph with no occurrences of $\overline{p}$ is produced. As a result, we have a spanned connection graph with at most n distinct atoms, and by the induction hypothesis, there is a refutation free of tautology creation by link. This proves

Theorem 2. Connection-graph resolution restricted to link activations that do not create tautologies by link is refutation complete for spanned ground connection graphs. □

3.4 Strong Completeness for Unit Preference Ordered Resolution

In [11] it is shown that ordered resolution is a special case of *cg*-resolution[7]; essentially the same can be said for ordered resolution with a unit preference strategy and limited use of subsumption. The key is Lemma 7 below. We begin with some basic definitions.

It is easy to see that some ordered links are present in any spanned connection graph since the path containing the maximal element from each clause must contain a link. We say that the connection graph $(\mathcal{S}, \mathcal{L})$ is *semi-full* if whenever a link is missing from $\mathcal{L}$, the *cg*-resolvent is in $\mathcal{S}$. In other words, every link missing from $\mathcal{L}$ has been activated. In this paper we are only concerned with connection graphs that are semi-full with respect to ordered links in the sense that the only missing links are ordered. Henceforward, the term semi-full implies that only ordered links are missing; in particular, any literal that is not the head of its clause is fully linked.

In [11] pure clauses arise only after the last link to a head literal is activated. As a result, such a clause could not subsequently participate in an ordered *cg*-resolution step. Thus there is no distinction between deleting such a clause as pure or ignoring it because ordered links to it can no longer exist. Implicit in [11] was retention of such clauses for bookkeeping purposes. In this paper we make it explicit: A clause that is pure because the head is pure will be retained. Such clauses may be deleted by *cg*-subsumption or if another literal becomes pure.

The next lemma says that *every* ordered link not yet activated must be present. It is essentially Lemma 5 from [11]; it is restated here using the notion of semi-full.

[7] This observation was made independently by Harald Ganzinger.

Lemma 7. Let $\mathcal{G}$ be a semi-full connection graph, and suppose that the literals of $\mathcal{G}$ have a total ordering. Suppose further that $\mathcal{H} = (\mathcal{S}, \mathcal{L})$ is obtained from $\mathcal{G}$ by an ordered *cg*-resolution step. Then $\mathcal{H}$ is semi-full.

Proof. Observe first that all resolvents of activated links are present: All were present in $\mathcal{G}$, and the resolvent of the activated link is in $\mathcal{S}$. Thus, if $\mathcal{H}$ is not semi-full, there is some ordered link $\{l_E, \overline{l}_F\}$ in $\mathcal{S}$ that is not in $\mathcal{L}$ and has not been activated. Since $\mathcal{G}$ was semi-full, this link must have been deleted in an earlier ordered *cg*-resolution step and inherited as deleted by the last resolution step. Hence, at least one of the two clauses E, F, say $E = \{l\} \cup E'$, was the resolvent of clauses C and D. Then l cannot be maximal in its parent, nor can any occurrence of it be maximal in any of its ancestors. Thus, the link $\{l_E, \overline{l}_F\}$ can never have been activated and hence could never have been deleted. □

The strategy of restricting *cg*-resolution to ordered links was adopted in [11] because strong completeness could be proved. Here unit preference is added because it provides a better inference rule — after all, it is almost always an advantage to activate links to unit clauses — and because it still admits a proof of strong completeness. The key is maintaining semi-fullness. Specifically, we will show that unit preference ordered *cg*-resolution will eventually yield a saturated connection graph that is semi-full. We then prove that, if the formula is unsatisfiable, the empty clause must be present.

Unit preference ordered cg-resolution is defined with the following additions to ordered *cg*-resolution. First, a clause that is pure because the head is pure is retained. Secondly, if a unit clause $\{p\}$ is present in a connection graph $\mathcal{G}$, then a *complete unit step* is performed using $\{p\}$: All clauses that are *cg*-subsumed by the unit are deleted from $\mathcal{G}$. Then $\{p\}$ is resolved against *every* clause to which it is linked. Each resolvent automatically *cg*-subsumes the other parent, and that parent is deleted. Note that if $\{p\}$ is fully linked, then it necessarily *cg*-subsumes every clause that contains the literal p, in particular, there is only one copy of $\{p\}$. In that case, a complete unit step will produce a connection graph with no occurrences of p (nor of $\overline{p}$). If the initial connection graph is full, then every unit clause is fully linked. The next lemma says that if semi-fullness is maintained during a deduction, unit clauses created by ordered *cg*-resolution steps will be fully linked.

Lemma 8. Let $\mathcal{G}$ be a semi-full connection graph, and suppose that the unit clause $\{p\}$ is the resolvent of an ordered *cg*-resolution step. Then $\{p\}$ is fully linked.

Proof. The literal p cannot be the head of either parent clause since the resolution step is ordered. Since the connection graph is semi-full, occurrences of p in the parents must be fully linked. Hence the inherited occurrence is fully linked. □

Lemma 9. Let $\{p\}$ be a fully linked unit clause in a semi-full connection graph. If a complete unit step is performed using $\{p\}$, then the resulting connection graph is semi-full.

Proof. Since p is fully linked, it cg-subsumes every clause that contains p. Deleting such clauses eliminates only links to literals in the deleted clauses, so the property of all non activated links being present is preserved. Thus we must show that if, say, the clauses A and B have been resolved upon — i.e., if their heads were linked but the link is not present — then the resolvent C is present. If it is not present, then it must have been subsumed by $\{p\}$. Then p must occur in A or in B, say in A. But then A is subsumed by $\{p\}$ and so would have been removed. Thus the connection graph produced by subsuming all clauses containing p is semi-full.

The proof that a semi-full c-graph results from resolving the unit $\{p\}$ with every clause containing $\overline{p}$ and then subsuming the parents is similar. □

The next two theorems prove that unit preference ordered *cg*-resolution is strongly complete.

Theorem 3. Let $\mathcal{G}$ be a connection graph with the full set of links and suppose that the literals of $\mathcal{G}$ have the atom ordering Then any sequence of unit preference ordered resolution steps in which tautologies are not used as parent clauses will eventually terminate with a semi-full connection graph.

Proof. An ordered *cg*-resolution step deletes the activated link. Any new link created by the step will precede the activated link, and thus the link multiset is smaller after the *cg*-resolution step. The same is true after any complete unit step since such a step eliminates links without adding any. Since the multiset ordering on link sets is well founded, termination is assured. To complete the proof, we must show that the resulting connection graph is semi-full.

The initial connection graph has the full set of links; in particular, it is semi-full. Lemmas 7 and 9 ensure that ordered *cg*-resolution steps and complete unit steps preserve semi-fullness. The only other operations, and thus the only operations that might produce a connection graph that is not semi-full, are applications of the pure rule. An ordered *cg*-resolution step can make pure only a head literal, and the pure rule is not applied to such clauses. Unit resolution steps also cannot produce pure clauses because the resolvents have the links from the parents, which are then removed.

The *cg*-subsumption steps can produce pure clauses. Deleting them will not change the property that any missing link has been resolved upon, so all that needs to be proved is that the resolvent of every activated link is present. So suppose that A and B are clauses whose heads were linked, and that that link is not present. We must show that the resolvent C is present. If not, i.e., if C has been deleted by application of the pure rule, then some literal, say q, other than the head of C is pure. Note that q occurs in one of the parents, say A. Since the head was resolved upon, q is not the head of A. Thus, before the subsumption deletions were made, q was fully linked in both A and C. Thus, the deletions would have made both occurrences of q pure. Hence, A would also be deleted by the pure rule. □

The proof method for the next theorem is due to Bachmair and Ganzinger [2]; the proof presented below is a refinement of an elegant exposition by Paliath

Narendran. An interpretation is described as a set of atoms, indicating that atoms in the set are assigned *true* and atoms not in the set are assigned $false$.

Theorem 4. Let $\mathcal{G} = (\mathcal{S}, \mathcal{L})$ be an unsatisfiable semi-full connection graph that is saturated with respect to unit preference ordered resolution. Suppose that the literals of $\mathcal{G}$ have the atom ordering. Then $\mathcal{S}$ contains the empty clause.

Proof. Suppose to the contrary that $\mathcal{S}$ does not contain the empty clause. Let $\mathcal{S} = \{C_1, C_2, ..., C_k\}$, where $C_i \prec C_j$ if $i < j$[8]. For each C_i, iteratively define an interpretation I_i, where $I_0 = \emptyset$, and the interpretation I_i is constructed from I_{i-1} as follows: $I_i = I_{i-1}$ if the maximal literal in C_i is negative or if I_{i-1} satisfies C_i. Otherwise, $I_i = I_{i-1} \cup \{q\}$ where q is the maximal literal in C_i. In that case, C_i is said to *produce* q. Let $I = I_k$.

Now, since $\mathcal{S}$ is unsatisfiable, there is a first clause C_i (necessarily non-empty) that is falsified by I. Then the head of C_i cannot be positive, since if it were, the fact that I falsifies the other literals in C_i would mean that I would have assigned *true* to the head. Let $\overline{p}$ be the head of C_i. Then $p \in I$, so there must be some earlier clause C_j that produced p. All other literals in C_j are $false$ since otherwise C_j would not have produced p. This implies that neither C_j nor C_i is a tautology; in particular, every non-head literal in both clauses precedes p and is different from p and is falsified by I. Thus, $\{p_{C_j}, \overline{p}_{C_i}\}$ is an ordered link that is not to a tautology. Hence, it must have been activated, and since $\mathcal{G}$ is semi-full, its resolvent C must be present. Moreover, C is falsified by I, and C precedes both C_j and C_i, contradicting the fact that C_i is the first clause falsified by I. □

Since the connection graph is semi-full at every stage in the proof, there is no difference between deleting ordered links upon activation and simply marking the links as used. As a result, unit preference ordered resolution and unit preference ordered *cg*-resolution are identical as inference rules at the ground level. The next theorem is an immediate corollary of the last two.

Theorem 5. Unit preference ordered resolution and unit preference ordered *cg*-resolution are strongly complete at the ground level. □

3.5 Remarks

Theorem 4 is based on the assumption of saturation. In combination with Theorem 3, it can be seen that, as with ordered *cg*-resolution alone, saturation criteria need never be tested: The supply of ordered links is simply guaranteed to run out, regardless of the order in which they are selected. Note also that the clauses retained for bookkeeping play only that role to enable the proof of Theorem 4. Once the head literal becomes pure, the clause can never be used in a subsequent inference. As a result, deleting the clause would not have any impact on the remaining steps in the deduction.

[8] Technically, $\mathcal{S}$ is a set of clause occurrences, but the proof is identical with or without clause repetitions as long as the clauses are listed in increasing order.

The strong completeness of ordered resolution depends on two properties: termination and the presence of the empty clause upon termination. Termination follows easily from the fact that activating an ordered link reduces the set of (not yet activated) ordered links in the multiset ordering, which is well-founded. However, if a *non-ordered* link to a unit clause is activated, the set of ordered links could increase in the multiset ordering. This problem is solved by deleting the subsumed parent. But that deletion creates a new problem: The deleted parent might be the resolvent of an ordered link, which in turn invalidates the argument in Theorem 4. That is, the clause C, which is the resolvent of clauses C_i and C_j, might not be present. Thus, the complete unit step is crucial for the proof if not for the result itself.

The reader may find a semantic tree viewpoint useful intuition. Consider the semantic tree $\mathcal{T}$ for a clause set $\mathcal{S}$ corresponding to a given atom ordering. A complete unit step involving clause $\{p\}$ corresponds to cutting out the level of $\mathcal{T}$ that contains p's and $\overline{p}$'s; pairs of branches that differ only in that one contains p and the other $\overline{p}$ are combined in the resulting tree $\mathcal{T}'$.

The tree $\mathcal{T}'$ is still closed for the resulting clause set $\mathcal{S}'$. Consider two branches $B_{\overline{p}}$ in $\mathcal{T}$ that are identical except that $B_{\overline{p}}$ contains $\overline{p}$, and B_p contains p. If there is a failure node above $\overline{p}$, then that failure node is in the same spot in $\mathcal{T}'$. Otherwise, the unit clause is falsified at $\overline{p}$. If B_p fails below p, so does the combined branch in $\mathcal{T}'$. If B_p fails at p, it fails on some clause $C_{\overline{p}}$ containing $\overline{p}$. Then the parent of $\overline{p}$ and p is an inference node in $\mathcal{T}$ and a failure node on the combined branch in $\mathcal{T}'$; it falsifies the unit resolvent of $\{p\}$ and $C_{\overline{p}}$.

The order in the tree is not changed, so the operation is compatible with ordered resolution steps. Moreover, the tree gets smaller, ensuring termination.

Theorem 5 provides an interesting strong completeness result. For connection graphs, the invariance of the semi-full link condition is crucial. A semi-full graph is one in which the only missing links are represented by the resolvents due to their activation. In other words, no links are not present due to the absence of inheritance, which is the important search pruning tool of unrestricted *cg*-resolution. Somehow, link deletion is limited to recording what has been explicitly done and thus essentially prohibits duplication of steps. Thus the progress for *cg*-resolution is limited but is potentially more significant for ordered resolution.

We conclude with a few remarks on the first-order case. In standard first-order resolution, a clause is unaffected when it becomes the parent of a resolvent, whereas in *cg*-resolution one of its links is deleted. This lifts only from ground proofs in which *every* instance of the activated link is deleted. Nevertheless, Theorem 4 seems to lift for ordered clause set resolution with little modification. Of course, unit resolutions cannot be pursued indefinitely as in the ground case, but must be alternated with rounds of ordered resolution in some fair way (see [16]). However, subsumptions by units and of parents by unit resolvents can be employed, since all instances of these operations would have been done in the ground proof.

References

1. Anderson, R., and Bledsoe, W., A linear format for resolution with merging and a new technique for establishing completeness, *J. ACM* 17(3), 525–534, 1970.
2. Bachmair, L. and Ganzinger, H., Rewrite-based equational theorem proving with selection and simplification, *Journal of Logic and Computation* 4(3), 217–247, 1994.
3. Bachmair, L. and Ganzinger, H., *Resolution Theorem Proving*, in Robinson, A. and Voronkov, A., editors, *The Handbook of Automated Reasoning*, chapter 2, volume I, 19–99, Elsevier Science Publishers, 2001.
4. Bibel, W., On matrices with connections, *J. ACM* **28**, 633–645, 1981.
5. Bibel, W., *Automated Theorem Proving*, 2^{nd} ed, Vieweg, Wiesbaden, 1987.
6. Bibel, W. and Eder, E., Decomposition of tautologies into regular formulas and strong completeness of connection graph resolution, *J. ACM*, **44** *2*, 320–344, 1997.
7. Brown, F., Notes on chains and connection graphs, Personal Notes, Dept. of Computation and Logic, University of Edinburgh, 1976.
8. Dershowitz, N. Termination of rewriting. In J.-P. Jouannaud, editor, *RTA-87*, pages 69–115. Academic Press, 1987. Reprinted from *J. of Symbolic Computation.*
9. Eisinger, N., What you always wanted to know about clause-graph resolution, In Jörg Siekmann, editor, *Proceedings* of *CADE-8*, Oxford, England, *LNAI* **230**, Springer-Verlag, 316–336, July 1986.
10. Eisinger, N., *Completeness, Confluence, and Related Properties of Clause Graph Resolution*, Research Notes in Artificial Intelligence, Pitman Publishing, 1991.
11. Hähnle, R., Murray, N.V. and Rosenthal, E. Ordered resolution vs. connection graph resolution. In R. Goré, A. Leitsch, eds., Proc. Int. Joint Conf. on Automated Reasoning, Siena, Italy. *LNAI* **2083**, Springer-Verlag, 182–194, June, 2001.
12. Joyner, W.H. Resolution strategies as decision procedures. *J. ACM* **23** *1*, 398-417, July 1976.
13. Kowalski, R. and Hayes, P.J., Semantic trees in automatic theorem proving, In *Machine Intelligence* **4**, Edinburgh University Press, 87–101, 1969. Reprinted in [22].
14. Kowalski, R. *Studies in the Completeness and Efficiency of Theorem-Proving by Resolution.* Ph.D. Thesis, University of Edinburgh, 1970.
15. Kowalski, R. A proof procedure using connection graphs, *J. ACM*, **22** *4*, 572–595, 1975.
16. Loveland, D.W., *Automated Theorem Proving: A Logical Basis*, North-Holland, New York, 1978.
17. Murray, N.V., and Rosenthal, E. Dissolution: Making paths vanish. *J. ACM* **40**, *3*, 504–535, July 1993.
18. Reynolds, J., Unpublished seminar notes, Stanford University, Palo Alto, CA, 1966.
19. Siekmann, J., Universal unification, In R.E. Shostak, editor, *Proc.* 7^{th} *Conference on Automated Deduction*, *LNCS* **170** , Springer-Verlag, 1–42, 1984.
20. Siekmann, J. and Stephan, W., Completeness and soundness of the connection graph procedure, Interner Bericht Institut I, Fakultät für Informatik, Universität Karlsruhe, 1980.
21. Siekmann, J. and Wrightson, G., editors, *Automation of Reasoning: Classical Papers in Computational Logic* **1**, *1957–1966*, Springer-Verlag, 1983.
22. Siekmann, J. and Wrightson, G., editors, *Automation of Reasoning: Classical Papers in Computational Logic* **2**, *1967–1970*, Springer-Verlag, 1983.
23. Siekmann, J. and Wrightson, G., An open research problem: Strong completeness of R. Kowalski's connection graph proof procedure, *Logic Journal of the IGPL* **10** *1*, 85–103, 2002.
24. Smolka, G., Completeness of the connection graph proof procedure for unit-refutable clause sets, In *Proceedings* of the 6^{th} *German Workshop on Artificial Intelligence*, Informatik Fachberichte 58, Springer-Verlag, Berlin, 191–204, 1982.
25. Smolka, G., Einige Ergebnisse zur Vollständigkeit der Beweisprozedur von Kowalski, Diplomarbeit, Fakultät für Informatik, Universität Karlsruhe, 1982.

26. Wos, L., Carson, D., and Robinson, G.A., The unit preference strategy in theorem proving, In *Fall Joint Computer Conference, AFIPS, Washington D.C.*, 615–621, Spartan Books, 1964. Reprinted in [21].
27. Wrightson, G. Strong completeness. Personal Notes, School of Math and Physics, Mcquarie University, NSW, Australia, 1987.